IN THE CLAIMS:

1. (Original) An apparatus for parsing an input data stream, comprising:

a first storage component operable to store a history buffer for containing an unencoded version of at least one previously encoded string;

a first comparison component operable to compare a string from said input data stream with said unencoded version of said at least one previously encoded string;

a second storage component operable to store: an indicator that there exist at least two matches found by said first comparison component, and tokens corresponding to said at least two matches;

a summing component operable to sum potential token lengths to provide total potential token lengths;

a second comparison component operable to compare said total potential token lengths;

a selection component operable to select a match corresponding to a shortest total token length to represent said string from said input data stream; and

an emitting component for emitting tokens representing said match corresponding to a shortest total token length.

2. (Original) An apparatus as claimed in claim 1, wherein said apparatus comprises a data compression apparatus.

3. (Original) An apparatus as claimed in claim 2, wherein said apparatus comprises an adaptive dictionary based data compression apparatus.

4. (Original) An apparatus as claimed in claim 3, wherein said apparatus comprises a Lempel-Ziv data compression apparatus.

5. (Original) An apparatus as claimed in claim 1, wherein said apparatus comprises a data encryption apparatus.

6. (Original) An apparatus as claimed in claim 1, wherein said apparatus comprises a message digest generation apparatus.

7. (Original) A method for parsing an input data stream, comprising:

storing a history buffer for containing an unencoded version of at least one previously encoded string;

comparing a string from said input data stream with said unencoded version of said at least one previously encoded string;

storing: an indicator that there exist at least two matches found by said first comparison component, and tokens corresponding to said at least two matches;

summing potential token lengths to provide total potential token lengths;

comparing said total potential token lengths;

selecting a match corresponding to a shortest total token length to represent said string from said input data stream; and

emitting tokens representing said match corresponding to a shortest total token length.

8. (Original) A method as claimed in claim 7, wherein said tokens comprise compressed data corresponding to said at least two matches.

9. (Original) A method as claimed in claim 8, wherein said compressed data comprises adaptive dictionary based compressed data.

10. (Original) A method as claimed in claim 7, wherein said tokens comprise encrypted data corresponding to said at least two matches.

11. (Original) A method as claimed in claim 7, wherein said tokens comprise message digest data corresponding to said at least two matches.

12. (Currently Amended) A computer program product, tangibly embodied in a computer-readable medium, comprising computer software code to perform, when the program element is executed ~~on~~ by a data ~~processing means~~ processor, operations comprising~~, the steps of~~:

storing a history buffer for containing an unencoded version of at least one previously encoded string;

comparing a string from said input data stream with said unencoded version of said at

least one previously encoded string;

storing: an indicator that there exist at least two matches found by said first comparison component, and tokens corresponding to said at least two matches;

summing potential token lengths to provide total potential token lengths;

comparing said total potential token lengths;

selecting a match corresponding to a shortest total token length to represent said string from said input data stream; and

emitting tokens representing said match corresponding to a shortest total token length.

13. (Currently Amended) A memory device storing computer program code to perform, when the program element is executed ~~on~~ by a data ~~processing means~~ processor, operations comprising~~;~~ ~~the steps of~~:

storing a history buffer for containing an unencoded version of at least one previously encoded string;

comparing a string from said input data stream with said unencoded version of said at least one previously encoded string;

storing: an indicator that there exist at least two matches found by said first comparison component, and tokens corresponding to said at least two matches;

summing potential token lengths to provide total potential token lengths;

comparing said total potential token lengths;

selecting a match corresponding to a shortest total token length to represent said string from said input data stream; and

emitting tokens representing said match corresponding to a shortest total token length.

14. (New) A parser comprising an input for receiving an input data stream comprising a string and an output, further comprising:

means for storing a history buffer having a content comprising an unencoded version of at least one previously encoded string;

means, having a first input coupled to said parser input and a second input coupled to said storing means, for comparing a string received from said input data stream with said history buffer content, said comparing means having an output coupled to said storing means for storing: (a) a flag for indicating that there exist at least two matches found by said comparing means and

(b) tokens corresponding to said at least two matches;

means for summing potential token lengths to output total potential token lengths;

means coupled to said summing means for selecting a shortest total token length to represent said string from said input data stream and having an output coupled to said parser output for outputting a corresponding token.

15. (New) A parser as in claim 14, further comprising data compression means based on an adaptive dictionary.

16. (New) A parser as in claim 15, where said data compression means comprises Lempel-Ziv data compression means.

17. (New) A method to parse an input data stream that comprises a string, comprising:

storing an unencoded version of at least one previously encoded string;

comparing a string received from the input data stream with the stored unencoded version of at least one previously encoded string to determine a case where there exist at least two matches;

determining tokens corresponding to the at least two matches;

summing potential token lengths to output total potential token lengths; and

outputting a token having a shortest total token length to represent the string from the input data stream.

18. (New) A computer program embodied on a computer readable medium for directing a computer to parse an input data stream that comprises a string, the computer program comprising program instructions the execution of which causes the computer to store an unencoded version of at least one previously encoded string; to compare a string received from the input data stream with the stored unencoded version of at least one previously encoded string to determine a case where there exist at least two matches; to determine tokens corresponding to the at least two matches; to sum potential token lengths to output total potential token lengths; and to output a token having a shortest total token length to represent the string from the input data stream.